Crack Detection in Concrete Surfaces using Image Processing, Fuzzy Logic, and Neural Networks

Gajanan K. Choudhary and Sayan Dey

***Abstract*—Automation in structural health monitoring has generated a lot of interest in recent years, especially with the introduction of cheap digital cameras. This paper presents fuzzy logic and artificial neural network based models for accurate crack detection on concrete. Features are extracted from digital images of concrete surfaces using image processing which incorporates the edge detection technique. The properties of extracted features are fed into the models for detecting cracks. Two kinds of approaches have been implemented in this study: the image approach which classifies an image as a whole, and the object approach which classifies each component or object in an image into cracks and noise. The models have been tested on 205 images and evaluated on the basis of five measures of performance.**

I. Introduction

MANY of the world's biggest and most important concrete structures are over decades old. As structures grow older, they grow weak and become prone to failures. Failure of high-rise buildings, dams, bridges, etc. results in tremendous loss of life and property, and hence, regular maintenance is required. Structural health monitoring, hence, has an important role to play during the operation phase of structures. Health monitoring also becomes a necessity during the construction phase for large-scale constructions.

A need for automation has been felt over the years as a large portion of construction depends heavily upon proper management, and management requires special skills and expertise in the respective fields, increasing the cost of construction, maintenance, as well as the time consumed for structural health monitoring and inspection. Automation also becomes necessary for inaccessible regions of structures. As far as health monitoring is concerned, acquiring data and automating the analysis of this data can go a long way in reducing the skilled labor requirement and the time consumption, thus reducing the cost of maintenance.

Automatic detection of cracks on various surfaces based on digital images is an area of active research. Most research in this field involves image processing and threshold based decision making. Tsao *et al.* (1994) [1] and Wang *et al.* (1998) [2] used this approach to detect defects in pavements. Kaseko *et al.* (1994) [3] implemented a neural network (NN) based model to classify pavement cracks which used crack tiles for detecting cracks.

Chae *et al.* (2001) [4] adopted a neuro-fuzzy approach to determine the health of sanitary sewer pipelines. The neural networks in the model were fed with pre-processed images as a whole and the networks identified the presence of various objects like cracks, joints etc. in the images and returned their attributes, which were then integrated using a fuzzy logic model to determine the overall health and condition of the pipes.

Khanfar *et al.* (2003) [5] proposed a non-destructive test using wavelets for identifying cracks through changes in the reflection coefficient of a surface. A fuzzy logic model was developed which used the reflection coefficient, frequency of operation and standoff distance to estimate the crack width and depth.

Byoung Jik Lee and Hosin "David" Lee (2004) [6] proposed crack classification based on crack tiles. Image-based, histogram-based and proximity-based neural networks were developed and compared.

A visual technique developed by Fujita *et al.* (2006) [7] involved preprocessing of images using subtraction of smooth images and a hessian matrix based line filter which then detected cracks by applying a threshold. This was further evolved by Fujita *et al.* (2009) [8] to include probabilistic relaxation and locally adaptive thresholding to improve the performance of the model.

A neuro-fuzzy classifier was developed by Sinha *et al.* (2006) [9] which involved fuzzifying the input feature vectors of all the objects in an image and then feeding them into a neural network module to classify pipe defects. The features used were area, major axis length, minor axis length, projections and number of objects in an approach.

Yagamuchi and Hashimoto (2008) [10] proposed a percolation-based model that considered the number of pixels and connectivity of the pixels to recognize cracks. . In this method, a central pixel was evaluated using a cluster generated by the percolation method.

Hyeong-Gyeong Moon and Jung-Hoon Kim (2011) [11] proposed a neural network with 5 hidden layers with the inputs being area and major to minor axes ratios of all the objects in an image at a time to determine the presence of cracks. The object properties are determined after image processing involving techniques of subtraction processing, Gaussian filtering, thresholding, morphological closing, and labeling.

In this research, an edge detection technique has been used instead of the subtraction processing and Gaussian filtering used in [11]. Four neural network models similar to

Manuscript received May 31, 2012, revised 21 September, 2012.

G. K. Choudhary is with the Department of Civil Engineering, Indian Institute of Technology-Kharagpur, Kharagpur, WB 721302 INDIA. (phone: 91-9735560083; E-mail: gajananchoudhary91@gmail.com)

S. Dey is with the Department of Civil Engineering, Indian Institute of Technology-Kharagpur, Kharagpur, WB 721302 INDIA (e-mail: syn.iitkgp@gmail.com).

978-1-4673-1744-3/12/$31.00 ©2012 IEEE

[11] have been trained, and the results have been compared with those of [11], to compare the effect of a different image processing method on the model performance. Further, an independent fuzzy logic model has also been developed and tested. Finally, 20 independent neural networks with sizes much lesser than and accuracies higher than the one in [11] have been developed, and their performance has been compared with Fujita *et al.* [8]. The general flowchart for crack recognition has been shown in Fig. 1.

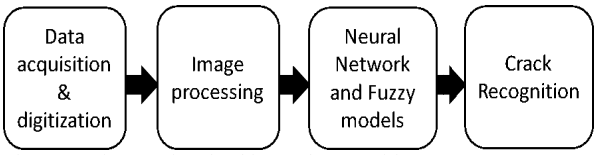


Fig. 1. Basic steps involved in crack recognition.

II. Data Acquisition

The data that was used for this research comprised of a total of 205 RGB images of various resolutions and aspect ratios. Most of the images were taken from the Structural Engineering Laboratory, Indian Institute of Technology Kharagpur, India, and some of the images were downloaded from the internet using Google image search.

The image of concrete should not contain any object other than concrete. Otherwise, due to the use of edge detection technique, the models detect the edges of these objects as cracks. Also, if the human eye itself cannot see the crack in the image, neither can any of the models in this research. The crack has to be at least visible no matter how much noisy the surface of the concrete is, only then is it possible to detect cracks using these models.

III. Image Processing

Any image contains irrelevant extra information which needs to be removed by preprocessing to facilitate the process of crack recognition by making it more efficient and time-saving. Various approaches in preprocessing of images have been used till now, of which the following steps have been adopted:

A. Resizing

All the images are resized to a size of 256 x 256 pixels. This step provides uniformity and saves computational time at the cost of loss of some information in the image. Also, with the kind of crack detection models that have been developed till now, resizing the image is a necessity for the models to work.

B. RGB-Grayscale Image Conversion

An image can be represented in terms of three color components: red (R), green (G), and blue (B). Each pixel in a RGB image is made up of an integer value of each of R, G, and B that lies between (and includes) 0 and 255.

In grayscale images, each pixel is represented with a single value between 0 and 255, where 0 corresponds to black and 255 to white. This value indicates the degree to which each pixel lights up, or the luminance (Y) of that pixel. The formula for conversion of images from RGB to grayscale is given by:

$$Y = 0.299\,R + 0.587\,G + 0.114\,B \qquad (1)$$

The images are converted from an RGB image to grayscale after resizing them. This is necessary to eliminate the effects of most of the color differences in the images.

C. Sobel Edge Detection

This method of edge detection is based on the calculation of the gradient in an image. This operation detects an edge wherever there is a steep gradient, i.e., a change in color or luminance.

A binary image is an image which is either strictly black or strictly white in color and can be represented as a matrix of 0's and 1's. All images are automatically converted to binary images by MATLAB [12] while detecting edges.

D. Morphological Operations

Several morphological operations are then performed on the detected edges for noise reduction and for joining some of the cracks that appear fragmented upon edge detection. Following are the steps involved:

1) Close: This is the morphological process by which small holes in the objects are filled. It is the process of dilation followed by erosion.

2) Bridge: Bridging sets '0' valued pixels to '1' if they have two nonzero neighbors that are not connected. This is a useful operation as using edge detection sometimes results in a fragmented crack. Some of these fragmented cracks are then joined with this operation if they are very close to each other.

This operation may also result in the joining of two nearby objects that are noise, but the pros of joining cracks that are fragmented far outweigh the cons of joining noise because joining results in a negligible increase in the area or ratio whereas joining two cracks drastically increases the area and ratio of the objects, which in turn increases the accuracy of the models.

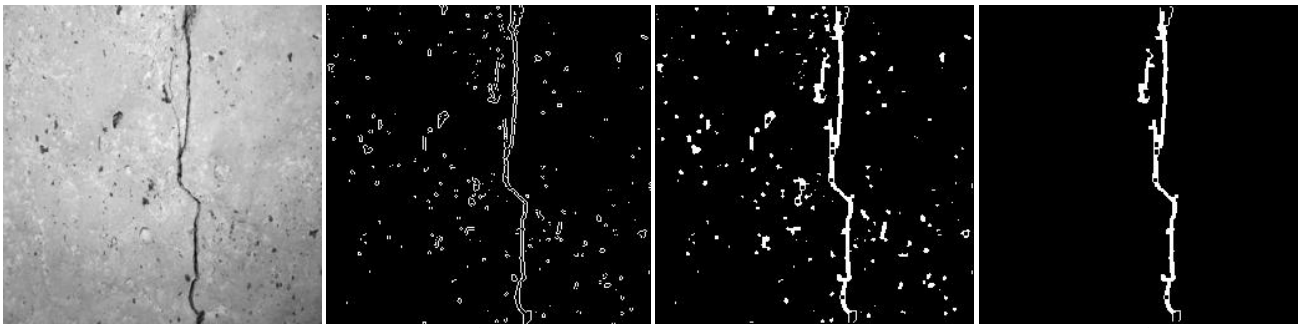
3) Spur: This removes individual pixels that are only diagonally connected at the border of an object in a binary image, called 'spur pixels.'

This process ends up removing pixels from cracked objects as well as noise, but the benefits of removing spur pixels from noise outweigh the disadvantages of removing those from cracks because removing the pixels results in a much larger percentage-wise reduction in the area of noise than of cracks. This, like bridging, is helpful for the models.

4) Clean: This process converts isolated 'on' pixels (single 1's surrounded by 0's) to zero. These objects are the easiest to classify as noise, so this step has been used only because it helps in noise reduction. This reduces the number of objects to be classified in the image (for the object approach) and saves computational time.

E. Connected-Component Labeling

In this operation, all the connected regions of an image are numbered serially. Labeling facilitates the identification of continuous, connected regions i.e., objects and their various properties in the next step.



(a) (b) (c) (d)

Fig. 2. (a) The original image after conversion to grayscale. (b) Feature extraction using edge detection technique. (c) Labeled image. (d) Detected crack.

F. *Object Properties*

The final step in image processing is to identify the object properties. These properties form the basis for crack detection using either fuzzy logic or neural networks. The choice of properties should be such that it enables and facilitates the differentiation between noise and cracks. Following are the properties chosen:

1) Area: The area of any object in an image is the total number of pixels that make up the object. The reason behind choosing this property is that in general, the area of cracks is higher than the area of noise.

2) Major-Minor Axes Ratio: The major and minor axes are the longest dimension of any object and the longest dimension perpendicular to the major axis at its midpoint such that the ellipse of these two dimensions as major and minor axes completely contains the given object. This property is chosen because the major-minor axes ratio is generally high for cracks as they are elongated whereas it is low for noise.

IV. General Approach for Model Development

One needs to understand that crack detection is essentially a classification problem. An image or an object in an image is considered and classified as a crack or noise. Reference [11] considers all the objects in an image at a time and feeds their properties to a neural network of a very large size. If each object is considered one at a time instead of all the objects, the problem reduces to a simple two-class classification based on very few input parameters (here, just 2 input parameters), thus drastically reducing the size of the neural network required and hence, the computational requirements and computational time.

There are two ways to handle the detection of cracks. The first ('Image approach,' Section VI-A) involves feeding all the data in an image to a neural network and getting a single output '0'or '1'depending on whether the image has at least one crack or not. This has been done in [11], but the image processing methods of [11] are different from those in this research. The effect of a different image processing method on the model performance has been compared with [11].

The second way is to consider each object separately and classify it as crack or noise ('Object approach'). This has been done for the fuzzy logic model (Section V) and another set of neural networks (Section VI-B). For testing of these models to be possible, one needs to first manually classify each object as a crack or noise, only then can the performance be evaluated. Since the 205 images chosen contain over 22000 objects in total, only the first 105 images have been chosen and the 12000 objects they contain have been manually classified as cracks or noise after visual inspection of the actual and processed images.

Since output of any of the models in this research is a value between 0 and 1, a decision boundary or a threshold (α) has to be determined in order to convert the output to a binary value. The image is classified as having at least one crack or no cracks depending on whether the value crosses or does not cross α respectively. Twomey *et al*. (2005) [13] showed that choosing 0.5 as α may not lead to accurate classification. In order to determine an optimum α, the decision boundary is varied from 0.01 to 0.99 and the threshold for which the model performs the best is chosen. This is explained in Section VII.

V. Fuzzy Logic Model

Fuzzy logic, introduced by L. A. Zadeh [14] in 1965 has found quite a place in modern industry as well as research. It is one of the few methodologies which are intuitive and can easily be understood by a layman since it uses easily understood linguistic variables.

For the fuzzy logic model, the input variables are 'Area' and 'Ratio' whereas the output variable is the 'Class' of the object. The range of each variable is between '0' and 'R' as shown in Fig. 3 and Table I. Each variable is made up of 2 or more fuzzy subsets and corresponding membership functions. The membership functions used are trapezoidal – 3 for 'Area,' 2 for 'Ratio,' and 2 for the output variable 'Class.' The membership functions have been chosen after

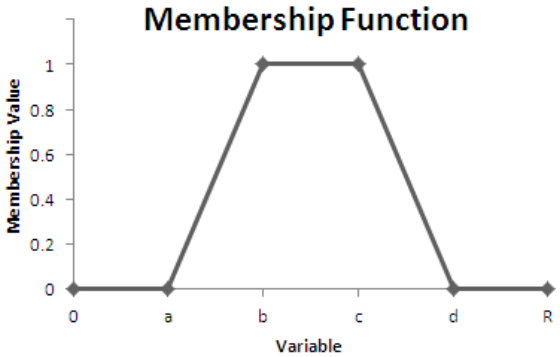


Fig. 3. Representation of Trapezoidal Membership Functions

TABLE I
MEMBERSHIP FUNCTIONS FOR THE FUZZY LOGIC MODEL

Variable	Variable Range (R)	Member-ship Function	Trapezoid definition			
			a	b	c	d
Area	5000	Low	0	0	50	100
		Moderate	50	100	150	500
		High	150	500	5000	5000
Ratio	200	Low	0	0	4.5	5.5
		High	4.5	5.5	200	200
Class	1	Noise	0	0	0.45	0.55
		Crack	0.45	0.55	1	1

plotting graphs of Ratio v/s Area, studying the relation between them and after trying various membership function intervals. It has been found that even though certain intervals of the subsets give a higher accuracy of prediction of at least one crack in an image, the overall number of cracks detected is lesser than some other ranges of the subsets of ratio. In this case, the detection of more number of cracks has been preferred over the image-wise crack detection accuracy of the model. The trapezoidal membership functions used can be represented by the four values 'a,' 'b,' 'c,' and 'd' as shown in Fig. 3, and their values have been given in Table I.

In the formulation of the fuzzy logic model, the Mamdani fuzzy inference system [15] has been used. Fuzzy inference rules are comprised of 'IF-THEN' statements, the 'IF' part is referred to as 'antecedent' whereas the 'THEN' part is called the 'consequent.' The antecedents can be combined by the fuzzy operators 'AND,' 'OR,' and 'NOT.' For this model, only the 'AND' operator is required. This operator uses the minimum weight or membership value of all the antecedents to restrict the membership value of the consequent to lie below this value. The rules have been given in Table II. A Mamdani 'min' operator [16] is used as the implication. The output of each rule is then aggregated by the 'max' operator which assigns the maximum occurring membership value in any of the rules to each of the output subsets (in this case, 'noise' and 'crack') at each point in the domain. Lastly, 'centroid' defuzzification is used, that is, the centroid of the aggregated fuzzy output is the crisp output.

TABLE II
FUZZY INFERENCE RULES

Rule No.	Area	Ratio	Class
1	High	High	Crack
2	High	Low	Crack
3	Moderate	High	Crack
4	Moderate	Low	Noise
5	Low	High	Noise
6	Low	Low	Noise

Example: IF 'Area' is 'High' AND Ratio is 'High,' THEN 'Class' is 'Crack.'

The crisp output has a value between 0 and 1, and is converted into a binary value using a decision boundary as explained in Section VII. The best model performance is at a threshold of 0.68 with an image-wise accuracy of 94.29%.

VI. NEURAL NETWORK MODELS

Two kinds of approaches have been implemented for the neural network models. The first one ('Image approach') involves feeding the area and ratio of all the objects of an image in the neural network and getting a single output – 1 or 0 depending on whether the image has cracks or not, respectively. The second approach ('Object Approach') involves feeding the area and ratio of only one object at a time to a neural network and getting a single output, classifying that object as a crack or noise.

Feed-forward back-propagation algorithm is used for training all the networks with training function 'trainscg' (scaled conjugate gradient) and learning function 'learngdm' [17]. The activation function is the log-sigmoid function. The networks are validated using the train-test-validation technique with a set of 105 (out of 205) images being divided into the three sets in the ratio of 0.7:0.15:0.15. The maximum number of epochs is set to 10000, and overtraining is prevented by stopping the training process of the networks when the error for the validation set increases for 30 consecutive iterations. Root mean square error (RMSE) is the error metric used in this study. Each neural network architecture is trained 50 times and the one with the least RMSE is selected.

A. Image Approach

In this approach, the object properties are sorted in the descending order of their areas. Since the number of objects is different for different images and the number of inputs to the neural network is fixed, for the sake of consistency, the sorted areas and major-minor axes ratios of only the top 50 objects in an image are given as the input to a multilayer neural network and it is determined if the image as a whole contains at least one crack or not. If an image has less than 50 objects, the rest of the input to the network is assigned '0.' Since each object is defined by two parameters, the input layer has a total of 50x2 = 100 nodes. There is a single output node with output '1' meaning the image has at least one crack and output '0' referring to an image without any crack. The number of hidden layers and number of hidden neurons has been varied, and the best architecture for a given number of hidden layers has been reported.

A neural network with 5 hidden layers (200-5-5-5-5-5-1 architecture) has been adopted in [11]. In this study, 4 types of neural networks containing 3, 4, 5, and 6 hidden layers respectively have been developed for comparison with [11]. Due to computational constraints, for a given number of hidden layers, the nodes in each of the hidden layers are kept equal and are varied in multiples of 5. The best networks selected on the basis of RMSE for 3, 4, 5, and 6 hidden layer networks are 100-50-50-50-1, 100-10-10-10-10-1, 100-5-5-5-5-5-1, and 100-5-5-5-5-5-5-1 respectively.

The decision boundary has to be selected on the basis of the image-wise accuracy after varying the threshold, and the best selected thresholds are 0.32, 0.41, 0.44 and 0.70 for 3, 4, 5, and 6 layer networks respectively.

B. Object Approach

In this approach, each object is classified individually as a crack or noise on the basis of its area and major-minor axes ratio. This reduces the problem to a two class classification problem in a two dimensional feature space. The input layer

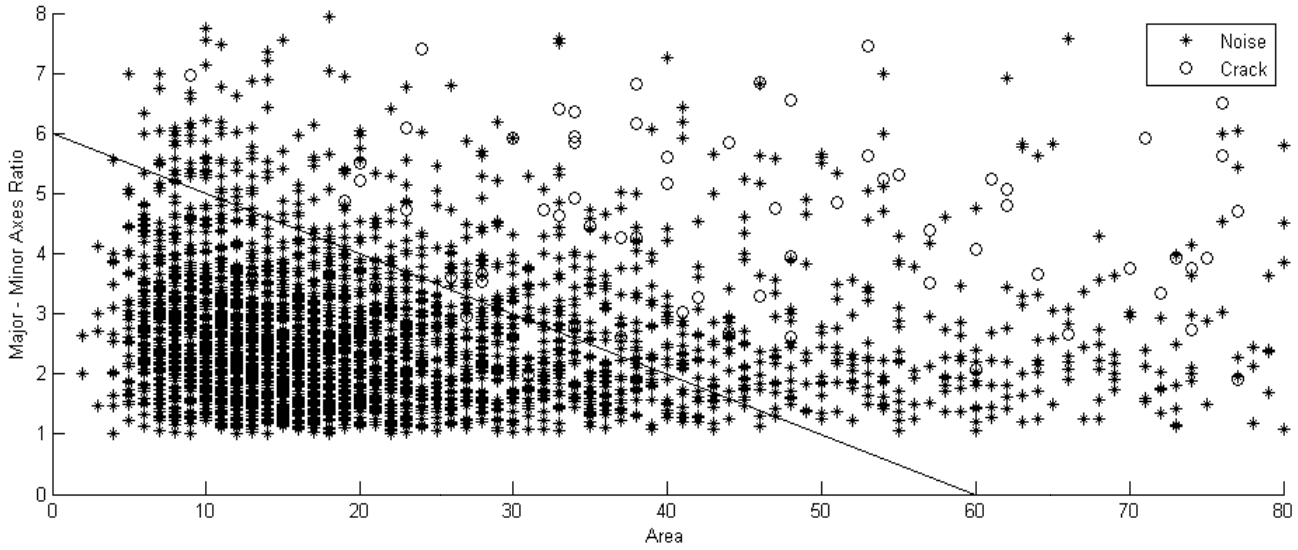


Fig. 4. Ratio v/s Area plot: The line drawn is the boundary between the region of noise and the transition region. This line is used for creating a balanced dataset. The actual range of the area goes up to 3500 and that of ratio goes up to 140.

has two nodes for the two parameters of the objects. Since the number of inputs is very less, a single hidden layer is used in this approach.

In almost every image, the number of objects which are cracks is far greater than the number of objects which are noise. This creates an imbalance in the training set which can make the model biased towards noise. A proper training dataset should be fairly balanced in terms of the numbers of each class and should also represent the complete dataset. As can be seen from Fig. 4, almost all the cracks lie above the imaginary line $6x + 60y = 360$. This line has been chosen at the approximate boundary between the region of noise and the transition region through visual inspection. (The regions have been explained in Section VIII-B.) Hence, the training set is created by including 100% of the cracks, 70% the noise lying above the line and 7.5% of all the noise lying below the given line. This is so because the concentration of noise in the low area and low major to minor axes ratio region is very high compared to the rest of the feature space.

A total of 20 neural networks have been trained by varying the number of nodes in the hidden layer from 1 to 20. The best neural network architecture ('2-13-1') is chosen after considering the 5 measures of model performance (explained in Section VII). The interpretive schedule is similar to the one used in the image approach. Fine-tuning of the threshold is then performed by varying it from 0.01 to 0.99 and two thresholds of 0.71 and 0.83 have been selected

VII. Experimental Results

Following are the measures of accuracy of the models (wherever applicable):

$$Image-wise\ Accuracy = \frac{\text{Images correctly classified}}{\text{Total number of images}} \quad (2)$$

$$Object-wise\ Accuracy = \frac{\text{Objects correctly classified}}{\text{Total number of objects}} \quad (3)$$

Fujita *et al*. [8] use the following criteria 3 for reporting their results:

$$Sensitivity = \frac{\text{Cracks correctly classified}}{\text{Total cracks}} \quad (4)$$

$$Specificity = \frac{\text{Noise correctly classified}}{\text{Total noise}} \quad (5)$$

$$Precision = \frac{\text{Cracks correctly classified}}{\text{Total objects classified as cracks}} \quad (6)$$

A. Performance of Fuzzy Logic Model

The graph of the image-wise accuracy of the fuzzy model v/s the decision boundary is shown in Fig. 5. From the graph, the value of the threshold can be chosen anywhere between 0.66 and 0.7. To further narrow down the choices, the sensitivity, specificity, and precision are compared in Table III and the value of the threshold has been finalized at 0.68 with an image-wise accuracy of 94.29%.

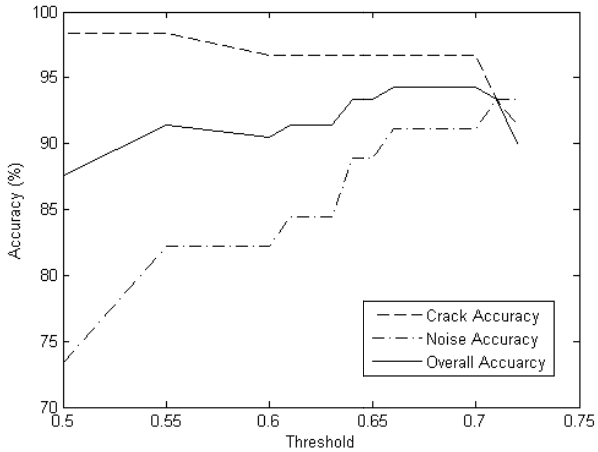


Fig. 5. Graph of image-wise accuracy of the fuzzy logic model with v/s threshold.

TABLE III
ACCURACY OF FUZZY MODEL FOR DIFFERENT VALUES OF THRESHOLD

Thres-hold	Sensit-ivity	Speci-ficity	Precis-ion	Image-wise Accuracy	Object-wise Accuracy
0.20	1.0000	0.0000	0.0200	0.5714	0.0200
0.30	0.5962	0.9943	0.6798	0.8000	0.9863
0.40	0.5423	0.9967	0.7705	0.8476	0.9876
0.50	0.5038	0.9976	0.8086	0.8762	0.9877
0.60	0.4654	0.9981	0.8345	0.9048	0.9874
0.70	0.4115	0.9985	0.8492	0.9429	0.9868
0.80	0.0000	1.0000	∞[a]	0.4286	0.9800
0.61	0.4615	0.9982	0.8392	0.9143	0.9874
0.62	0.4538	0.9982	0.8369	0.9143	0.9873
0.63	0.4500	0.9982	0.8357	0.9143	0.9872
0.64	0.4423	0.9983	0.8456	0.9333	0.9872
0.65	0.4346	0.9983	0.8433	0.9333	0.9871
0.66	0.4308	0.9983	0.8421	0.9333	0.9870
0.67	0.4269	0.9984	0.8473	0.9333	0.9870
0.68	0.4192	0.9984	0.8450	0.9429	0.9868
0.69	0.4115	0.9985	0.8492	0.9429	0.9868
0.70	0.4115	0.9985	0.8492	0.9429	0.9868
0.71	0.4115	0.9985	0.8492	0.9429	0.9868
0.72	0.4038	0.9986	0.8537	0.9429	0.9867
0.73	0.4000	0.9986	0.8525	0.9429	0.9866
0.74	0.3962	0.9986	0.8512	0.9429	0.9865
0.75	0.2577	0.9990	0.8375	0.8857	0.9841
Suggested model threshold and corresponding accuracy:					
0.68	0.4192	0.9984	0.8450	0.9429	0.9868

Accuracy has been reported for 105 images which have also been used to train the neural networks.

[a]All objects are classified as noise resulting in a zero in the denominator in (6) and hence, the infinite precision.

B. Performance of Neural Network Models

For the 4 neural networks based on image approach (Section VI-A), the sensitivity, specificity, precision, and object-wise accuracy cannot be checked as the models classify the image as a whole instead of the objects. Hence, the thresholds are varied and the best model is selected using only the image-wise accuracy of the model. These come out to be 0.32, 0.41, 0.44 and 0.70 for 3, 4, 5, and 6 layer networks with accuracies of 90.48%, 87.62%, and 87.62%, and 92.38% respectively for 105 images.

TABLE IV
VARIATION OF THE 5 ACCURACY MEASURES WITH THE NUMBER OF HIDDEN NODES IN THE NEURAL NETWORK MODELS FOR THRESHOLD = 0.5

No. of hidden nodes	Sensit-ivity	Speci-ficity	Precis-ion	Image-wise Accuracy	Object-wise Accuracy
1	0.1923	0.9997	0.9259	0.8667	0.9835
2	0.2692	0.9991	0.8642	0.8952	0.9845
3	0.5692	0.9959	0.7400	0.8286	0.9874
4	0.5731	0.9960	0.7450	0.8286	0.9875
5	0.5615	0.9966	0.7725	0.8571	0.9879
6	0.5731	0.9966	0.7760	0.8476	0.9881
7	0.5615	0.9965	0.7684	0.8667	0.9878
8	0.5808	0.9965	0.7704	0.8571	0.9881
9	0.5654	0.9962	0.7538	0.8286	0.9876
10	0.5577	0.9974	0.8146	0.8952	0.9886
11	0.5731	0.9965	0.7680	0.8571	0.9880
12	0.5577	0.9970	0.7923	0.8762	0.9882
13	0.5808	0.9966	0.7784	0.8667	0.9883
14	0.5769	0.9964	0.7653	0.8667	0.9880
15	0.5731	0.9969	0.7884	0.8762	0.9884
16	0.5769	0.9965	0.7692	0.8571	0.9881
17	0.5615	0.9966	0.7725	0.8667	0.9879
18	0.5654	0.9969	0.7903	0.8571	0.9883
19	0.5731	0.9964	0.7641	0.8571	0.9879
20	0.5615	0.9969	0.7892	0.8762	0.9882

Best architecture '2-13-1' is selected on the basis of highest sensitivity.

For the 20 neural networks in the object approach (Section VI-B), the best neural network of 2-13-1 architecture has been first chosen after considering the variation of the 5 measures of performance of the networks with a variation in the number of hidden nodes for various thresholds, one of which (threshold = 0.5) has been given in Table IV. For this network, the threshold is then fine-tuned and two network thresholds, one based on a reasonable sensitivity and good image-wise accuracy (threshold = 0.71), and one based on highest image-wise accuracy (threshold = 0.83) have been suggested in Table V.

C. Comparison with Earlier Work

It has been observed that sensitivity, specificity and precision alone (as reported in [8]) are not enough for the evaluation of the model performance, as is evident from the results given in Table VI. The threshold can be chosen in such a way that the sensitivity, specificity, and the precision of the model are comparable to that of [8]. For such a threshold, the number of objects that are noise, but have been wrongly classified as cracks, increases without a significant decrease in the specificity due to the extremely high number of objects that are noise. Hence, for such thresholds, the image-wise accuracy goes down significantly as more images which do not have any cracks get classified incorrectly. This can be seen for the 2-13-1 architecture neural network model in Table VI where the sensitivity, specificity, and precision are comparable to those reported in [8], but the model itself does not perform well since the image-wise accuracy is much lower at 73.33% than the best suggested model which has an accuracy of 96.19%.

On the contrary, for a higher threshold of 0.83 for the

TABLE V
ACCURACY OF NEURAL NETWORK WITH '2-13-1' ARCHITECTURE FOR DIFFERENT VALUES OF THRESHOLD

Thres-hold	Sensit-ivity	Speci-ficity	Precis-ion	Image-wise Accuracy	Object-wise Accuracy
0.65	0.5462	0.9975	0.8161	0.8857	0.9884
0.66	0.5462	0.9975	0.8161	0.8857	0.9884
0.67	0.5423	0.9975	0.8150	0.8857	0.9884
0.68	0.5385	0.9977	0.8284	0.8952	0.9885
0.69	0.5308	0.9980	0.8466	0.9048	0.9887
0.70	0.5269	0.9980	0.8457	0.9048	0.9886
0.71	0.5269	0.9981	0.8509	0.9143	0.9887
0.72	0.5231	0.9982	0.8553	0.9238	0.9887
0.73	0.5115	0.9983	0.8581	0.9238	0.9885
0.74	0.5077	0.9983	0.8571	0.9238	0.9884
0.75	0.5000	0.9983	0.8609	0.9238	0.9884
0.76	0.4962	0.9984	0.8658	0.9238	0.9884
0.77	0.4885	0.9984	0.8639	0.9238	0.9882
0.78	0.4885	0.9985	0.8699	0.9238	0.9883
0.79	0.4885	0.9987	0.8819	0.9333	0.9884
0.80	0.4808	0.9987	0.8865	0.9429	0.9884
0.81	0.4808	0.9988	0.8929	0.9524	0.9884
0.82	0.4692	0.9988	0.8905	0.9524	0.9882
0.83	0.4538	0.9990	0.9008	0.9619	0.9881
0.84	0.4462	0.9991	0.9063	0.9619	0.9880
0.85	0.4423	0.9991	0.9055	0.9524	0.9879
Suggested Thresholds for the 2-13-1 Neural Network Architecture:					
0.71	0.5269	0.9981	0.8509	0.9143	0.9887
0.83	0.4538	0.9990	0.9008	0.9619	0.9881

TABLE VI
COMPARISON OF RESULTS WITH FUJITA *ET AL.* [8] WITH THE PROPOSED '2-13-1' ARCHITECTURE NEURAL NETWORK MODEL

Thres-hold	Sensit-ivity	Speci-ficity	Precis-ion	Image-wise Accuracy	Object-wise Accuracy
Sample threshold for the proposed model:					
0.25	0.792	0.989	0.592	0.733	0.985
Best results reported by Fujita *et al.* [8]:					
-	0.722	0.993	0.655	Global Thresholding	
-	0.815	0.922	0.147	Local Thresholding	
-	0.805	0.992	0.631	Probabilistic Relaxation	

neural network architecture '2-13-1' given in Table V, the sensitivity drops down to a seemingly low value of about 45%, but the image-wise accuracy increases to 96%. It is important to note that this is because during image processing, some of the cracks get fragmented during the edge detection step. A part of these fragments may get classified as cracks and the remaining part as noise (whereas all such fragments are classified as cracks during the manual classification of the 12000 objects in the 105 images), thus resulting in reduced sensitivity.

Hence, from the two examples mentioned above, a seemingly low sensitivity (especially for the models developed in this research) *does not imply* that the model is inaccurate whereas a very high sensitivity along with specificity as high as 99% *does not imply* that the model is accurate.

The last step in this research is to test the developed models on a completely different set of images. This has been done on 100 new images. Sensitivity, specificity, precision, and object-wise accuracy have not been measured as they require manually classifying all the objects in all the images (which amount to nearly 10000 in these 100 images) as cracks or noise. Hence, only the image-wise accuracy has been checked and reported in Table VII. It can be inferred from the comparison with the results of [11] in Table VII that edge detection performs as good as other approaches such as median subtraction, Gaussian filter etc.

TABLE VII
IMAGE-WISE ACCURACY FOR FINAL TESTING WITH 100 IMAGES AND COMPARISON WITH MOON *ET AL.* [11]

Approach Type	Model	Threshold	Image-wise % Accuracy
Image	3 Hidden Layer NN	0.32	87.00
Image	4 Hidden Layer NN	0.41	86.00
Image	5 Hidden Layer NN	0.44	85.00
Image	6 Hidden Layer NN	0.70	93.00
Object	'2-13-1' NN	0.83	96.00
Object	Fuzzy Logic Model	0.68	94.00
Image	NN of [11]	-	90.25

VIII. DISCUSSION

A. Image Processing

It should be noted is that all the models developed in this research are valid exclusively when the resized resolution of the image is 256 x 256 pixels. The membership functions in the fuzzy logic model will have different ranges and new neural networks will have to be trained if the resized resolution of the image is different from the chosen one.

The main problem in image processing is that some of the cracks get fragmented into 2 to 3 smaller objects, resulting in a smaller area and ratio and thus, resulting in a wrong detection of some of the fragments as noise which in turn results in the reduction of the sensitivity of the model. A better way of joining these fragments needs to be implemented than the 'bridging' step mentioned in Section III-D, because bridging joins objects separated by only a single pixel whereas the crack fragments may be separated by as many as 10 pixels.

Lastly, the edge detection technique ends up detecting drastic color changes as edges – hence, it is suggested that instead of directly converting the image into a grayscale one and using edge detection, the three components R, G, and B should also be used separately along with the grayscale image to detect cracks. (Note that in an RGB image, if we take only one of the three of R, G, or B components, we get an image similar to a grayscale image.)

B. Object Parameters

The Ratio v/s Area plot can be roughly divided into 3 regions – the first region is where objects are certainly cracks, the second is where objects are certainly noise and the third is the transition region between the former two regions. When the object properties lie in the former two regions, it is fairly easy to classify them, and as expected, almost all the incorrect classifications lie in the transition region.

Basically, what we are feeding the neural network in the object approach is the area-ratio plot during training, and then asking it to classify a new object. Indeed, if the area-ratio plot were to be given to a human, the person would not be able to classify the object based on just the area and ratio when the area-ratio point of the object is in the transition region. It is possible that area and ratio alone are not good enough to determine whether an object is a crack or noise, since the sensitivity of all the models is low. Hence, we need at least one more parameter to correctly differentiate cracks from noise even in the transition region. It may be possible that for an additional parameter (say circularity [10]), the transition region between that parameter and area or ratio is different from the transition region of area and ratio. Such a parameter would then enable one to increase the sensitivity of the model significantly (though the increase in image-wise accuracy would not be high).

C. Accuracy

The accuracies of the models have been measured in five ways: the number of images detected correctly (image-wise accuracy), the number of objects detected correctly (object-wise accuracy), sensitivity, specificity, and precision.

It should be noted that neither the image-wise accuracy, nor the object-wise accuracy is an entirely correct measure of the accuracy of the models. In the approach using image-wise accuracy, if an image has multiple cracks, then even if only a single crack out of many in the image is detected, or even if noise is wrongly classified as a crack for that image, the image may still be correctly classified as one having a crack, but for a wrong reason. In the second approach of

object-wise accuracy, the accuracy of the model is extremely high (98.81% for 2-13-1 architecture, Table V) because most of the objects are noise which is relatively easy to detect whereas the number of cracks is very few, giving a false impression that the model performance is good. At the same time, just reporting the sensitivity, specificity and precision is also not enough as can be seen from the explanation in Section VII-C and Table VI.

Hence, all the 5 measures should be taken into account while evaluating the performance of any crack detection model. It should be noted that one may need to compromise on one of the measures to choose the best model, as can be seen from the difference in accuracies of the 2-13-1 network for two different thresholds given in Table V. Hence, one needs to fine-tune the model to suit the needs of the situation using different choices of threshold. For example, when one needs a high sensitivity and is ready to compromise slightly on the image-wise accuracy, one may choose a threshold of 0.3 for the fuzzy model, or if a one needs a high image-wise accuracy and needs to detect at least one crack in a given image, then one may use the suggested threshold of 0.68 for the fuzzy model (as is evident from Table III).

IX. Conclusion

A fuzzy logic model and various neural networks for detecting cracks and noise in images based on image and object approaches were developed and compared in this research. The performance of the models that used feature extraction based on edge detection was compared with earlier work in this field, and it was found that edge detection performs just as good as median subtraction, low pass Gaussian filter, etc., and that the object approach is better than the image approach. It was also found that neural networks outperform the fuzzy logic model in all measures of model performance. It is suggested that while reporting the performance of any model for crack detection, the 5 measures of accuracy from (2) to (6) should be considered together.

Acknowledgment

We would like to thank Prof. S. V. Barai for introducing the concepts of fuzzy logic and neural networks to us and for his support and guidance throughout the project. We also wish to thank Venkatesh S., Ved M., and Prerit Varun for their contribution to our work.

References

[1] S. Tsao, N. Kehtarnavaz, P. Chan, and R. Lytton, “Image-based expert-system approach to distress detection on CRC pavement,” *Journal of Transportation Engineering*, vol. 120, no. 1, pp. 62–64, January - February 1994.

[2] K. C. Wang, S. Nallamothu, and R. P. Elliott, “Classification of pavement surface distress with an embedded neural net chip,” *Artificial Neural Networks for Civil Engineers: Advanced Features and Applications*, pp. 131–161, January-February 1998.

[3] M. S. Kaseko, Z. P. Lo, and S. G. Ritchie, “Comparison of traditional and neural classifiers for pavement-crack detection,” *Journal of Transportation Engineering*, vol. 120, no. 4, pp. 552–569, July - August 1994.

[4] Chae M., and Abraham D., (2001). ”Neuro-Fuzzy Approaches for Sanitary Sewer Pipeline Condition Assessment.” *Journal of Computing in Civil Engineering 15, Special issue: Information technology for life-cycle infrastructure management*, Pages 4–14. doi: 10.1061/(ASCE)0887-3801(2001)15:1(4).

[5] Aws Khanfar, Mohammed Abu-Khousa, and Nasser Qaddoumi, “Microwave near-field non-destructive detection and characterization of disbonds in concrete structures using fuzzy logic techniques,” *Composite Structures*, Volume 62, Issues 3–4, Pages 335-339, ISSN 0263-8223, 10.1016/j.compstruct.2003.09.033, 2003.

[6] Byoung Jik Lee, and Hosin “David” Lee, “Position-Invariant Neural Network for Digital Pavement Crack Analysis,” *Computer-Aided Civil and Infrastructure Engineering,* Volume 19, Issue 2, pages 105–118, March 2004.

[7] Y. Fujita, Y. Mitani and Y. Hamamoto, “A Method for Crack detection on a Concrete Structure,” *18th International Conference on Pattern Recognition*, Volume 3, pp. 901–904, 2006.

[8] Y. Fujita, and Y. Hamamoto, “A robust method for automatically detecting cracks on noisy concrete surfaces,” *Next-Generation Applied Intelligence. 22nd International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems IEA/AIE 2009*, pp. 76–85, June 2009, Tainan, Taiwan.

[9] S. K. Sinha, P. W. Fieguth, and M. A. Polak, “Computer vision techniques for automatic structural assessment of underground pipes,” *Computer-Aided Civil and Infrastructure Engineering*, vol. 18, no. 2, pp. 95–112, February 2003.

[10] T. Yamaguchi, and S. Hashimoto, “Improved percolation-based method for crack detection in concrete surface images,” *19th International Conference on Pattern Recognition*, December 2008.

[11] H. Moon, and J. Kim, “Inteligent Crack Detecting Algorithm On The Concrete Crack Image Using Neural network,” *Proceedings of the 28th ISARC*, Pages 1461-1467, Seoul, Korea, 2011.

[12] The MathWorks, Inc., *Image Processing Toolbox for Use with MATLAB: User’s Guide*, August 2004.

[13] J. M. Twomey and A. E. Smith, “Validation and Verification,” *19th Artificial Neural Networks for Civil Engineers: Fundamentals and Applications*, Chapter 4. January 2005.

[14] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, pp. 338-353, 1965.

[15] J. -S. R. Jang and N. Gulley, *MATLAB Fuzzy Logic Toolbox: User’s Guide*, April 1997.

[16] E. H. Mamdani, "Applications of Fuzzy Set Theory to Control Systems: A Survey," in *Fuzzy Automata and Decision Processes*, M. M. Gupta, G. N. Saridis and B. R. Gaines, eds., North-Holland, New York, pp. 1-13, 1977.

[17] M. H. Beale, M. T. Hagan and H. B. Demuth, *Neural Network Toolbox™: User’s Guide,* September 2012.